**Семинар 5. РЕШАЮЩИЕ ДЕРЕВЬЯ ДЛЯ ЗАДАЧИ КЛАССИФИКАЦИИ**

**from** sklearn.tree **import** DecisionTreeClassifier

**from** sklearn.datasets **import** make\_blobs

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

%matplotlib inline

RANDOM\_SEED = 139

train\_data, train\_labels = make\_blobs(n\_samples=200, centers=[(0,1),(-3,-3),(4,2)],

n\_features=2, random\_state=RANDOM\_SEED,

cluster\_std=(1.2,1.5,1,))

# Let’s write an auxiliary function that will return grid for further visualization.

**def** **get\_grid**(data):

x\_min, x\_max = data[:, 0].min() - 1, data[:, 0].max() + 1

y\_min, y\_max = data[:, 1].min() - 1, data[:, 1].max() + 1

**return** np.meshgrid(np.arange(x\_min, x\_max, 0.01), np.arange(y\_min, y\_max, 0.01))

clf\_tree = DecisionTreeClassifier(criterion='entropy', max\_depth=3,

random\_state=RANDOM\_SEED)

# training the tree

clf\_tree.fit(train\_data, train\_labels)

# some code to depict separating surface

xx, yy = get\_grid(train\_data)

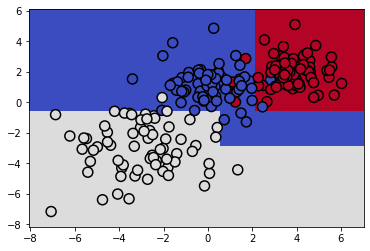
predicted = clf\_tree.predict(np.c\_[xx.ravel(),

yy.ravel()]).reshape(xx.shape)

plt.pcolormesh(xx, yy, predicted, cmap='coolwarm')

plt.scatter(train\_data[:, 0], train\_data[:, 1], c=train\_labels, s=100,

cmap='coolwarm', edgecolors='black', linewidth=1.5);



Задание 5.2.2

По графику выше определите, какова максимальная глубина этого решающего дерева? (Это сбалансированное дерево, т. е. все его ветви одинаковой глубины)

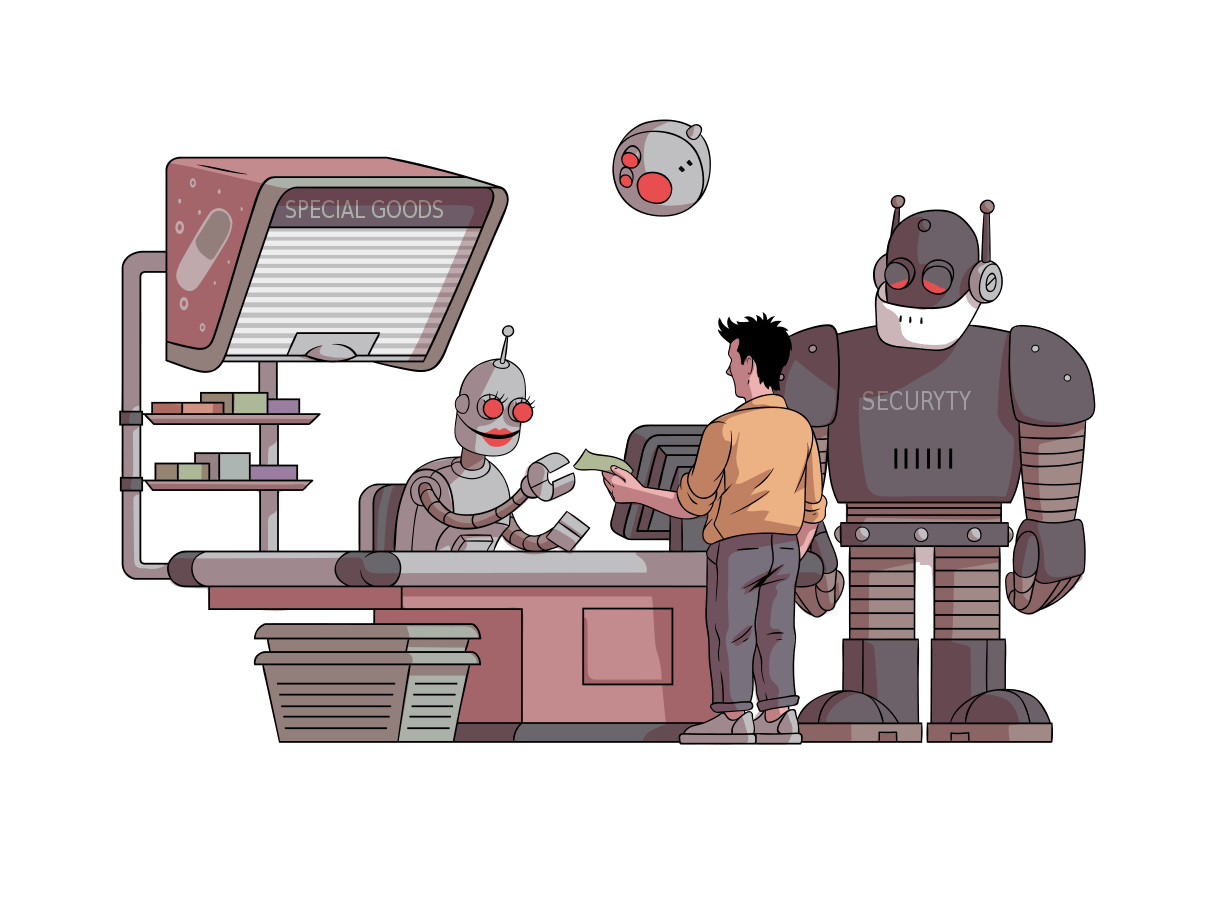
2

3

4

**ЗАДАНИЕ НА РЕАЛЬНОМ ДАТАСЕТЕ**

После тренировки на искусственно сгенерированных данных попробуем применить решающее дерево на реальных данных.



Вам предложен датасет с некоторыми характеристиками банкнот, по которым мы будем определять, является ли банкнота фальшивой или настоящей.

Параметры решающего дерева, которые понадобятся для решения задачи:

* max\_depth — максимальная глубина дерева.
* max\_features — максимальное число признаков, по которым ищется лучшее разбиение в дереве. Это нужно потому, что при большом количестве признаков будет «дорого» искать лучшее (по критерию типа прироста информации) разбиение среди *всех* признаков.
* min\_samples\_leaf — минимальное число объектов в листе. У этого параметра есть понятная интерпретация: если он равен 5, то дерево будет порождать только те классифицирующие правила, которые верны как минимум для 5 объектов.

Задание 5.2.3

1 point possible (graded)

Обучите на предложенных данных решающее дерево. Целевой переменной здесь является переменная Class. Размер тестовой выборки возьмите за 0.2, random\_state = 17 для разбиения и дерева. Максимальную глубину дерева примите за , максимальное число признаков, по которым ищется лучшее разбиение в дереве — за . Какое значение *f1-score* вы получили? Округлите до трёх знаков после точки-разделителя.  нет ответа

Задание 5.2.4

Проклассифицируйте банкноту с вектором признаков 2.04378,-0.38422,1.437292,0.76421. К какому классу она относится?  нет ответа

Здесь приведена **визуализация** того, как одно решающее дерево разделяет выборку, состоящую из **трёх классов**.

Видно, что решающее дерево может очень неплохо отделить каждый класс от всех остальных. Разделяющая поверхность каждого класса кусочно-постоянная, и при этом каждая сторона поверхности параллельна оси координат, так как каждое условие сравнивает значение ровно одного признака с порогом.

В то же время решающее дерево вполне может переобучиться: его можно сделать настолько глубоким, что каждый лист решающего дерева будет соответствовать ровно одному объекту обучающей выборки. В этом случае, если записать в каждом листе ответ соответствующего объекта, на обучающей выборке получается нулевая ошибка. Дерево получается явно **переобученным**. Пример такого дерева:

RANDOM\_SEED = 139

train\_data, train\_labels = make\_blobs(n\_samples=100, centers=[(-3,-3),(4,2)],

n\_features=2, random\_state=RANDOM\_SEED,

cluster\_std=(5,5))

clf\_tree = DecisionTreeClassifier(criterion='entropy', max\_depth=None,

random\_state=RANDOM\_SEED)

# training the tree

clf\_tree.fit(train\_data, train\_labels)

# some code to depict separating surface

xx, yy = get\_grid(train\_data)

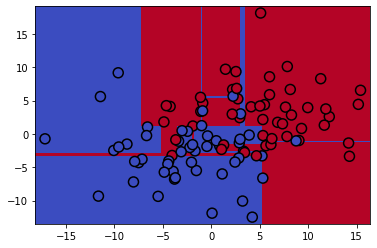
predicted = clf\_tree.predict(np.c\_[xx.ravel(),

yy.ravel()]).reshape(xx.shape)

plt.pcolormesh(xx, yy, predicted, cmap='coolwarm')

plt.scatter(train\_data[:, 0], train\_data[:, 1], c=train\_labels, s=100,

cmap='coolwarm', edgecolors='black', linewidth=1.5);



**РЕШАЮЩИЕ ДЕРЕВЬЯ В ЗАДАЧЕ РЕГРЕССИИ**

**from** sklearn.tree **import** DecisionTreeClassifier

**from** sklearn.datasets **import** make\_blobs

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

%matplotlib inline

RANDOM\_SEED = 139

n\_train = 150

n\_test = 1000

noise = 0.1

**def** **f**(x):

x = x.ravel()

**return** np.exp(-x \*\* 2) + 1.5 \* np.exp(-(x - 5) \*\* 2)

**def** **generate**(n\_samples, noise):

X = np.random.rand(n\_samples) \* 10 - 5

X = np.sort(X).ravel()

y = np.exp(-X \*\* 2) + 1.5 \* np.exp(-(X - 5) \*\* 2) + \

np.random.normal(0.0, noise, n\_samples)

X = X.reshape((n\_samples, 1))

**return** X, y

X\_train, y\_train = generate(n\_samples=n\_train, noise=noise)

X\_test, y\_test = generate(n\_samples=n\_test, noise=noise)

**from** sklearn.tree **import** DecisionTreeRegressor

reg\_tree = DecisionTreeRegressor(max\_depth=4, random\_state=RANDOM\_SEED)

reg\_tree.fit(X\_train, y\_train)

reg\_tree\_pred = reg\_tree.predict(X\_test)

plt.figure(figsize=(10, 6))

plt.plot(X\_test, f(X\_test), "b")

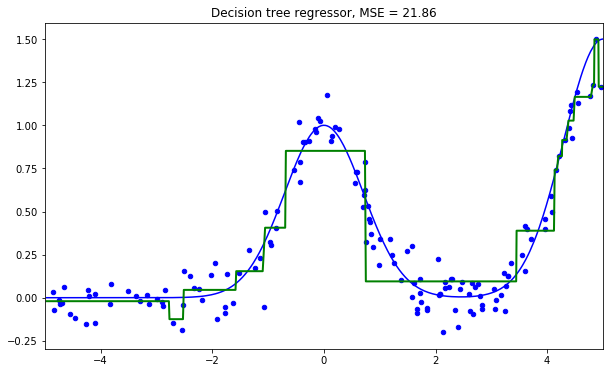
plt.scatter(X\_train, y\_train, c="b", s=20)

plt.plot(X\_test, reg\_tree\_pred, "g", lw=2)

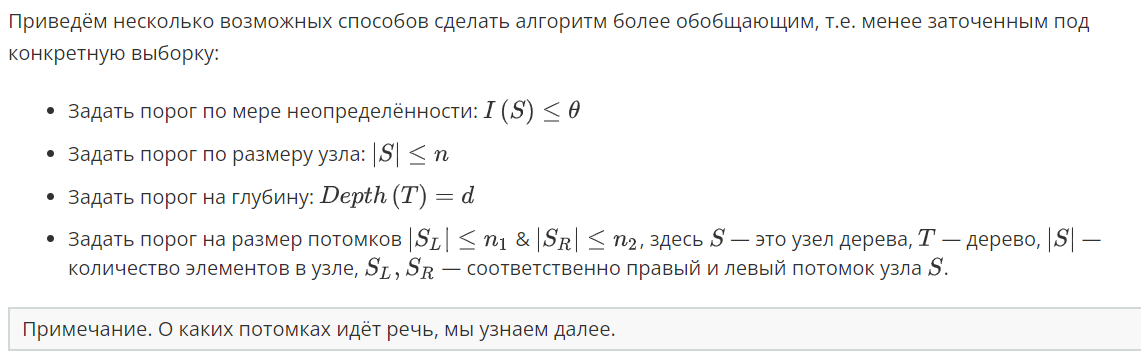
plt.xlim([-5, 5])

plt.title("Decision tree regressor, MSE = %.2f" % np.sum((y\_test - reg\_tree\_pred) \*\* 2))

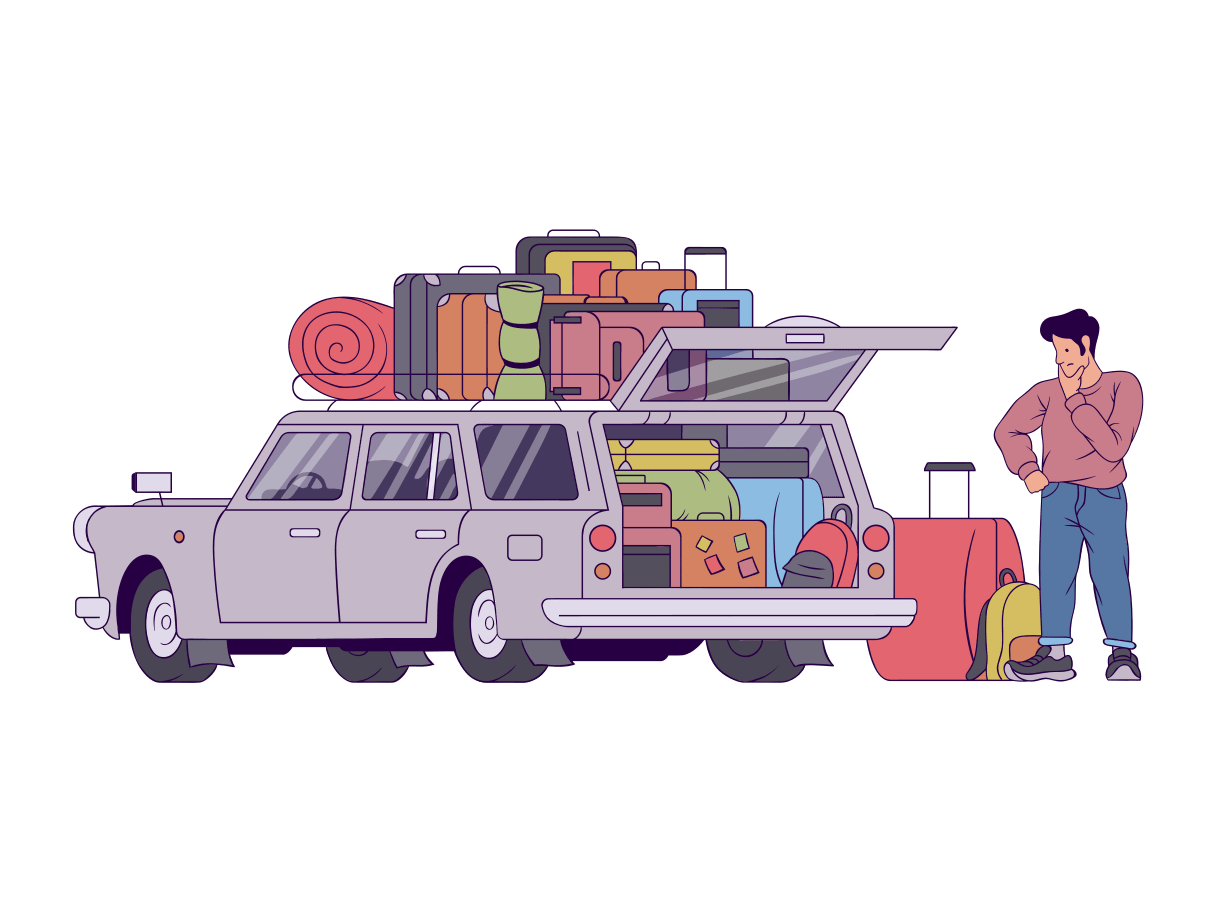
plt.show()



**БОРЬБА С ПЕРЕОБУЧЕНИЕМ (РЕГУЛЯРИЗАЦИЯ)**



**ЗАДАЧА О ПОТРЕБЛЕНИИ ТОПЛИВА**



Потренируемся реализовывать задачу регрессии с помощью решающих деревьев на реальных данных. В данной задаче мы попробуем предсказать потребление топлива.

Задание 5.2.5

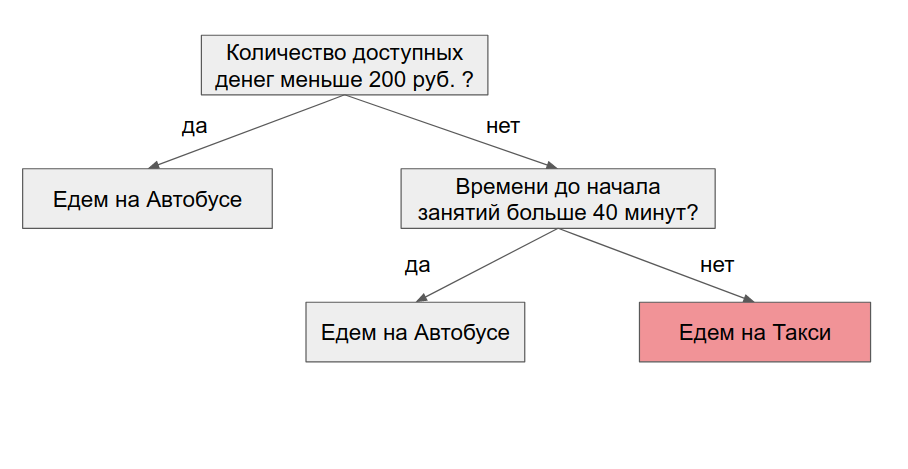
Обучите решающее дерево для регрессии на предложенных данных, размер тестовой выборки возьмите за 0.3, random\_state = 42 для разбиения и дерева. Вычислите *RMS*E, округлите до двух знаков после точки-разделителя.

Какова глубина дерева?

Задание 5.4.1

1 point possible (graded)

На рисунке представлено решающее дерево «Ситуации из жизни» с признаками «количество доступных денег» и «количество времени до важного события».



Пусть дана следующая выборка из пяти объектов (первый признак — деньги, второй — время):

[250, 45]

[100, 35]

[400, 30]

[250, 60]

[300, 50]

Сколько из них попадёт в красный лист?